

**HOME AUTOMATION USING ESP32 MICROCONTROLLER
AND
BLYNK APP**

By

ABHISHA CHAKRABARTI- 22BEC1187

RALF PAUL VICTOR- 22BEC1222

VINAY- 22BEC1247

MRINANK GAUR- 22BEC1258

MANAN MALIK- 22BEC1245

A project report submitted to

Dr. Ravi Prakash Dwivedi

Associate Professor

SCHOOL OF ELECTRONICS ENGINEERING

in partial fulfilment of the requirements for the course of

BECE304P – Analog Communication Systems

Lab Slot: D2+TD2

in

**B. Tech. ELECTRONICS AND COMMUNICATION
ENGINEERING**



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Vandalur – Kelambakkam Road

Chennai – 600127

APRIL 2024

BONAFIDE CERTIFICATE

Certified that this project report entitled “**Home Automation using ESP32 and Blynk App**” is a bonafide work of **Abhisha Chakrabarti (22BEC1187), Ralf Paul Victor (22BEC1222), Vinay (22BEC1247), Mrinank Gaur (22BEC1258) and Manan Mallick (22BEC)**, who carried out the Project work under my supervision and guidance for **BECE304P- Analog Communication Systems**

Dr. Ravi Prakash Dwivedi

Associate Professor (Grade-I)

School of Electronics Engineering (SENSE),

VIT Chennai

Chennai – 600 127.

ABSTRACT

The assimilation of Internet of Things (IoT) technology into everyday existence has facilitated inventive resolutions across multiple fields, such as **home automation**. The goal of this project is to create a home automation system using the **Blynk mobile application and the ESP32 microcontroller**.

The ESP32 functions as the main control unit for the home automation system and has Wi-Fi connection and a large processing capacity. Its strong points allow it to communicate with other smart gadgets and sensors that are placed all throughout the house with ease. Key features of the proposed home automation system include remote control of lighting, temperature regulation, security monitoring, and appliance management. Users can remotely monitor and adjust these parameters in real-time, enhancing convenience, energy efficiency, and security within the home.

Operating remotely via a smartphone or tablet, the Blynk app serves as the user interface, offering simple management over the appliances and gadgets that are linked. Control interfaces and automation logic may be easily customized because to its user-friendly interface. The project is based on both hardware and software development. The hardware part incorporates the connection of the ESP32 Module with the relay (electronic switch) and the LEDs and motors used as Home-appliances for the project. The software part includes writing the code on Arduino IDE to connect it to the ESP32 module and the Blynk App. The Blynk app acts as the user interface, providing intuitive control over the connected devices and appliances remotely from a smartphone or tablet.

Thus, this IoT based project represents a paradigm shift in how we interact with the physical world, empowering us to create smarter, more connected, and efficient systems that enhance productivity, improve quality of life, and enable sustainable development. This project helps us to realise the potential of IoT in residential environment. Technology has made things easier for us, increasing efficiency of modern household.

ACKNOWLEDGEMENT

We express our sincere gratitude and heartfelt thanks to our guide, **Dr Ravi Prakash Dwivedi Sir**, School of Electronics Engineering, for this consistent encouragement and invaluable guidance throughout the course of the project.

His expertise in the field of Analog Communication System has been instrumental in shaping the direction of this project. We are truly grateful for the time and effort he has dedicated to mentoring us. He has mentored us in the right way possible with insightful feedback and suggestions that have greatly enriched this project.

NAME: Abhisha Chakrabarti

NAME: Mrinank Gaur



NAME: Ralf Paul Victor

NAME: Vinay



NAME: Manan Malik

TABLE OF CONTENTS

SERIAL NO.		TITLE	PAGE NO.
		ABSTRACT	
		ACKNOWLEDGEMENT	
1		INTRODUCTION	6
	1.1	OBJECTIVE AND GOAL	7
	1.2	APPLICATION	7
	1.3	FEATURE	8
2		DESIGN AND IMPLEMENTATION	9
	2.1	BLOCK DIAGRAM	10
	2.2	HARDWARE ANALYSIS	10
	2.3	(SNAPSHOTS-PROJECT, TEAM, RESULTS)	12
3	3.1	SOFTWARE –CODING AND ANALYSIS	14
	3.2	(SNAPSHOTS OF CODING AND RESULTS)	22
4		CONCLUSION AND FUTURE WORK	23
	4.1	RESULT, CONCLUSION AND INFERENCE	23
	4.2	FUTURE WORK	23
5		REFERENCES	25
6		PHOTOGRAPH OF THE PROJECT ALONG WITH THE TEAM MEMBERS	26

1 INTRODUCTION

In recent years, there has been a significant rise in the adoption of smart home technologies, enabling homeowners to control various devices and systems with ease and convenience. Home automation systems have revolutionized the way we interact with our living spaces, providing enhanced comfort, energy efficiency, and security. In this report, we will explore the concept of home automation using ESP32, relay modules, and the Blynk app.

The ESP32, a powerful microcontroller module, has gained immense popularity due to its versatility and compatibility with various IoT applications. By leveraging the capabilities of the ESP32, combined with relay modules and the Blynk app, homeowners can create a smart home ecosystem that can be controlled remotely from their smartphones or tablets.

By combining the ESP32, relay modules, and the Blynk app, homeowners can enjoy a wide range of benefits. Firstly, they can remotely control and monitor their home appliances and devices, enabling them to turn lights on or off, adjust room temperatures, or even activate security systems from anywhere in the world. This level of control not only enhances convenience but also contributes to energy conservation by optimizing the usage of electrical resources.

Secondly, home automation systems provide an added layer of security. With the ability to monitor security cameras, lock or unlock doors, and receive real-time alerts, homeowners can ensure the safety of their homes even when they are away. This peace of mind is invaluable in today's fast-paced world.

Lastly, home automation systems offer scalability and expandability. With the ESP32 as the central hub, additional sensors, actuators, or modules can be integrated into the system to extend its capabilities. This allows homeowners to gradually upgrade and customize their smart homes according to their evolving needs and preferences.

In conclusion, home automation using ESP32, relay modules, and the Blynk app opens a world of possibilities for homeowners to create a smarter, more efficient, and secure living environment. By harnessing the power of these technologies, individuals can take control of their homes like never before. In the following sections of this report, we will delve deeper into the technical aspects of implementing such a system and explore various use cases and potential challenges.

1.1 OBJECTIVES AND GOALS

The objective of this project is to design and implement a cost-effective and user-friendly home automation system. The system will allow users to control various home appliances remotely using a simple web-based interface. The ESP32 microcontroller will serve as the heart of the system, interfacing with the relay modules to control the appliances.

Goals:

- **Develop a robust and reliable hardware setup:** The project aims to create a well-designed hardware configuration utilizing the ESP32 microcontroller and relay module. This includes properly connecting and configuring the components to ensure seamless communication and control.
- **Implement a user-friendly mobile interface:** The goal is to integrate the Blynk app, enabling users to interact with the home automation system through an intuitive and visually appealing mobile interface. The app should facilitate effortless control and monitoring of connected devices.
- **Enable remote control and scheduling:** The system should allow users to remotely turn appliances on or off, adjust their settings, and even schedule specific operations. This empowers users to manage their home appliances efficiently, saving energy and enhancing convenience.
- **Ensure security and privacy:** The project strives to implement robust security measures to protect the home automation system from unauthorized access. This includes encryption protocols, user authentication, and secure data transmission, safeguarding the privacy and integrity of user information.

1.2 APPLICATION

- **Convenience and comfort:** Users can conveniently operate their home appliances from anywhere within the Wi-Fi range. For example, they can turn on the air conditioner before arriving home, ensuring a comfortable environment upon arrival.
- **Energy efficiency:** By remotely controlling appliances, users can minimize energy wastage. They can turn off lights or appliances accidentally left on, adjust temperature settings, and create schedules for efficient energy consumption.
- **Enhanced security:** The system can be integrated with security features such as motion sensors, door/window sensors, and surveillance cameras. Users can receive real-time notifications and remotely monitor their homes, improving overall security.
- **Accessibility and inclusivity:** Home automation enables individuals with mobility challenges or disabilities to control their environment more easily. With the Blynk app, users can interact with the system using accessible interfaces and voice commands.

1.3 FEATURE

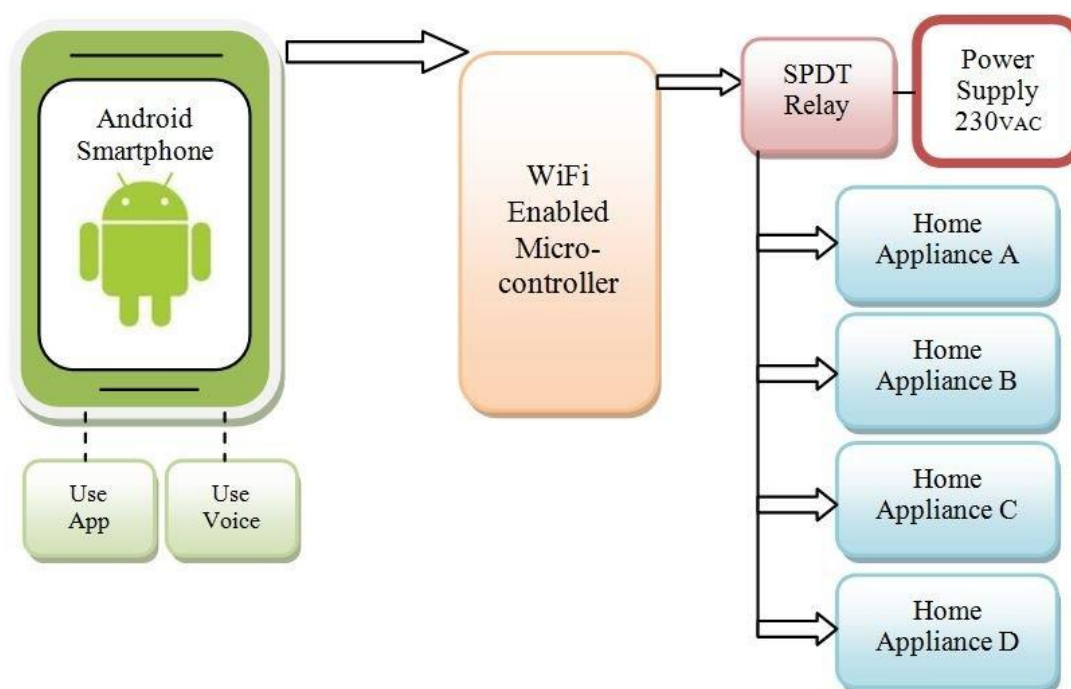
- **Remote Control:** Users can remotely control their home appliances through the Blynk app, providing convenience and flexibility. They can turn devices on/off, adjust settings, and monitor their status from anywhere with an internet connection.
- **Customizable Interface:** The Blynk app offers a customizable interface, allowing users to design their own control panels with buttons, sliders, and other interactive elements. This feature provides a personalized and intuitive user experience.
- **Notifications and Alerts:** Users can receive notifications and alerts on their smartphones or tablets regarding the status of their appliances. For instance, they can be notified if a door is left open or if a device is malfunctioning.
- **Integration with Sensors:** The system can be integrated with various sensors such as motion detectors, temperature sensors, and humidity sensors. This integration enables automation based on environmental conditions, enhancing comfort and efficiency.
- **Security and Privacy:** Robust security measures are implemented to protect the home automation system from unauthorized access. This includes secure communication protocols, encrypted data transmission, and user authentication.
- **Voice Control:** The system can be integrated with voice assistants like Amazon Alexa or Google Assistant, allowing users to control their appliances using voice commands. This feature adds an extra layer of convenience and accessibility.

2 DESIGN AND IMPLEMENTATION

In the design, the Esp 32 microcontroller is connected to the 4-channel relay module (5v relay module) through the data pins of the Esp –32 microcontroller, From the Esp –32 microcontroller the signal is passed to the relay module through these data pins to the signal pin of the relay module, which control the relay module (ON/OFF of the relay module), and the whole thing is connected to 5v power supply , which is given to the Esp-32 module and relay module which is operated on 5v

The Esp-32 module is Wi-Fi operated module which is connected the phone through wi-fi and through blynk app, the signal is passed to the module through Wi-Fi, and when signal is given to microcontroller the Esp –32 read the signal and give command according to the signal to relay module , and when the relay module get the signal it switch on , and the appliance/ load is get on . After that if we send off signal to the microcontroller through blynk app, and again it goes to the Esp through Wi-Fi, is again give signal to the relay module and then the relay module gets switched off, and the load connected to the relay module is also gotten switch off

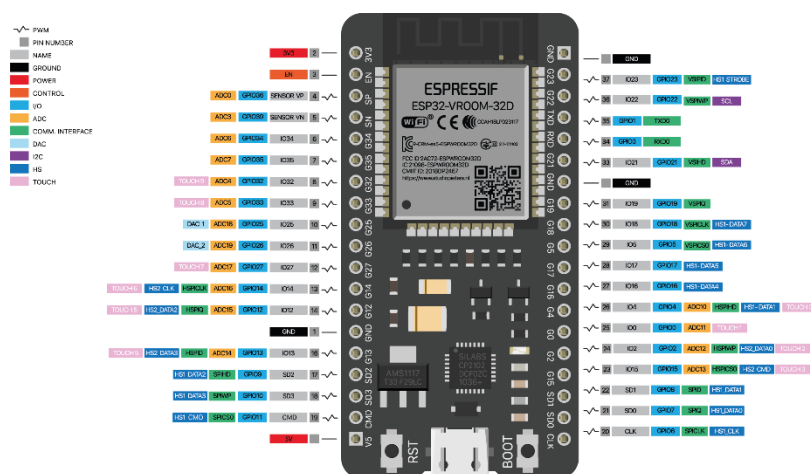
2.1 BLOCK DIAGRAM



- Here the Wi-Fi enabled module is Esp-32 which is connected to the same Wi-Fi network as it connected to the phone, then it gives command to esp-32 and then the esp-32 give signal to the relay module and relay module controlled the home appliance

2.2 HARDWARE ANALYSIS

■ ESP-32



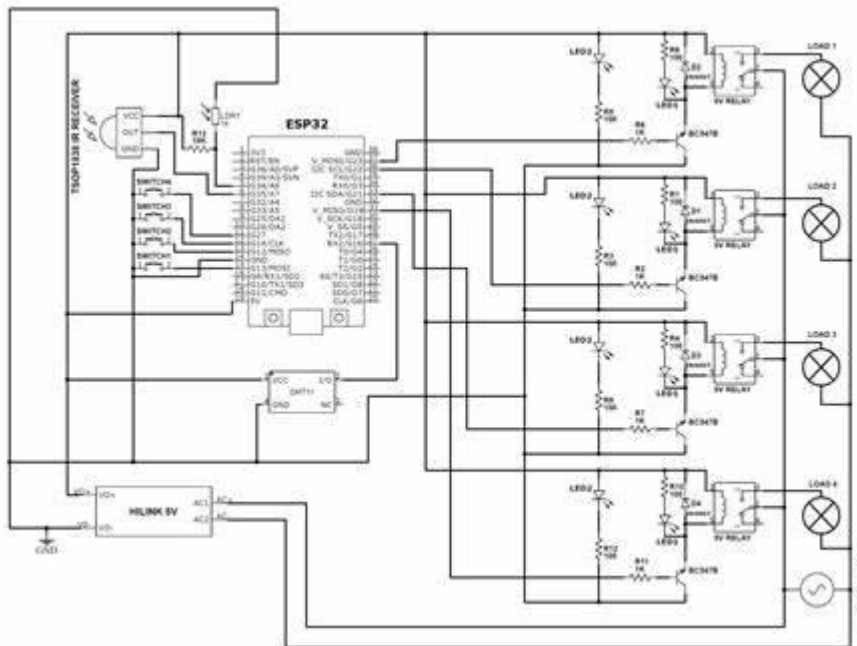
According to Alexandrea et al., (2017), The ESP32 is a low-cost, low-power system on a chip series of microcontrollers with Wi-Fi and Bluetooth capabilities and a highly integrated structure powered by a dual-core Tensilica Xtensa LX6 microprocessor. is a dual-core system with two Harvard

■ 5v-Relay Module



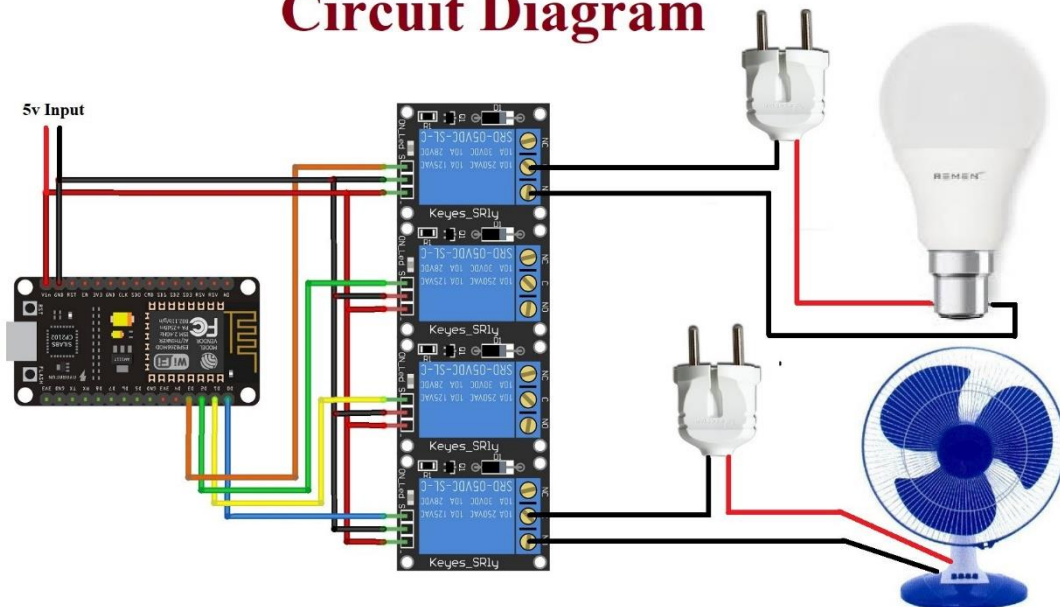
The 5V relay module generally includes a coil, and two contacts like normally open (NO) and normally closed (NC)4. The relay coil is energized by DC so that contact switches can be opened or closed4. A 5v relay is an automatic switch that is commonly used in an automatic control circuit and to control a high-current using a low-current signal4. The input voltage of the relay signal ranges from 0 to 5

- Schematic diagram



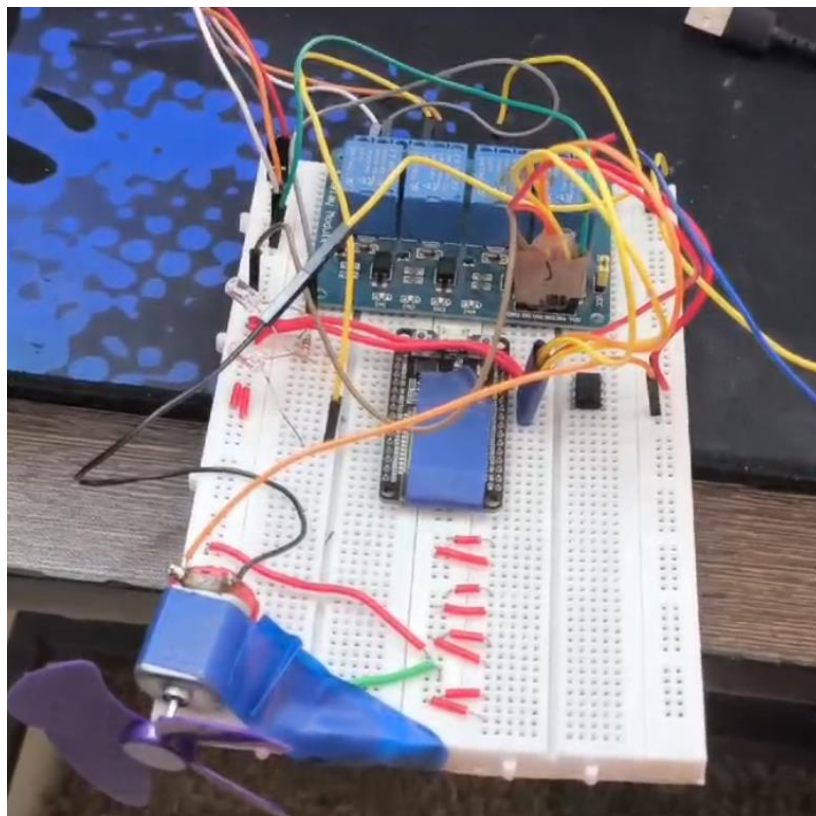
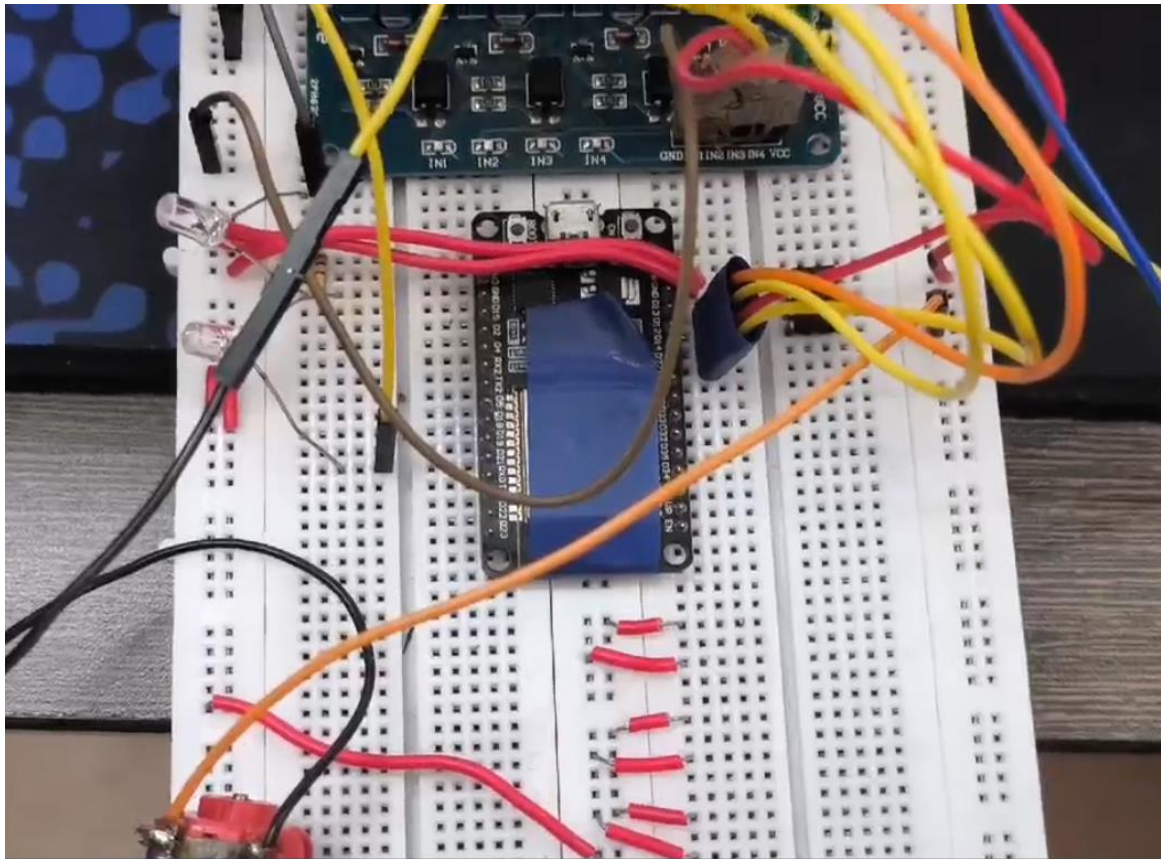
- Circuit diagram

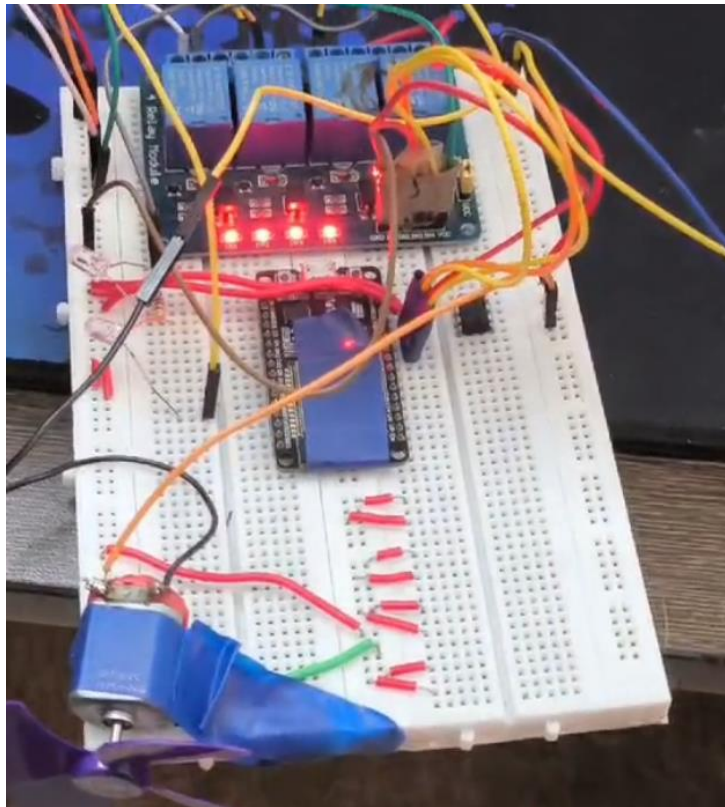
Circuit Diagram



Here is the circuit diagram of the project, the 4 digital pins are connected to the signal pins of the 4-channel relay module and input voltage supply is connected to the 5V supply which is given to the ESP-32 and relay module, by this circuit diagram the home automation system is prepared.

2.3 SNAPSHOTS-PROJECT, TEAM, RESULTS





3 SOFTWARE

In this project we are using Blynk Iot software as an interface between phone and esp-32 module, Blynk is a low-code IoT software platform that enables the connection of devices to the cloud¹²³. It allows users to build mobile and web applications to remotely control and monitor these devices¹²³. Blynk is designed for both personal IoT projects and commercial connected products, empowering users to analyse real-time and historical data from devices². It also facilitates the management of thousands of users and deployed products in a secure cloud

For the programming of the esp-32 microcontroller, here we use Arduino IDE to program the code in the Esp-32 microcontroller. The Arduino IDE is very user friendly and easy to use IDE which is widely used to programme Board like Arduino and esp-32, esp8266, it is very convenient to use and we can easily programme the board by using it

3.1 CODING AND ANALYSIS

```
#define BLYNK_TEMPLATE_ID "ENTER_TEMPLATE_ID"
#define BLYNK_DEVICE_NAME "ENTER_DEVICE_NAME"
#define BLYNK_AUTH_TOKEN "ENTER_AUTH_TOKEN"
```

```
// Comment this out to disable prints and save space
```

```
#define BLYNK_PRINT Serial
```

```
#include <WiFi.h>
```

```
#include <WiFiClient.h>
```

```
#include <BlynkSimpleEsp32.h>
```

```
char auth[] = BLYNK_AUTH_TOKEN;
```

```
// Your WiFi credentials.
```

```
// Set password to "" for open networks.
```

```
char ssid[] = "ENTER_WIFI_SSID";
```

```
char pass[] = "ENTER_WIFI_PASSWORD";
```

```
BlynkTimer timer;
```

```
#define button1_pin 26
```

```
#define button2_pin 25
```

```
#define button3_pin 33
```

```
#define button4_pin 32
```

```
#define relay1_pin 13
```

```
#define relay2_pin 12
```

```
#define relay3_pin 14
```

```
#define relay4_pin 27
```

```
int relay1_state = 0;
```

```
int relay2_state = 0;
```

```
int relay3_state = 0;
```

```
int relay4_state = 0;
```

```
//Change the virtual pins according the rooms
```

```
#define button1_vpin  V1
```

```
#define button2_vpin  V2
```

```
#define button3_vpin  V3
```

```
#define button4_vpin  V4
```

```
//-----

// This function is called every time the device is connected to the Blynk.Cloud

// Request the latest state from the server

BLYNK_CONNECTED() {

  Blynk.syncVirtual(button1_vpin);

  Blynk.syncVirtual(button2_vpin);

  Blynk.syncVirtual(button3_vpin);

  Blynk.syncVirtual(button4_vpin);

}

//-----

// This function is called every time the Virtual Pin state change
//i.e when web push switch from Blynk App or Web Dashboard

BLYNK_WRITE(button1_vpin) {

  relay1_state = param.asInt();

  digitalWrite(relay1_pin, relay1_state);

}

//-----

BLYNK_WRITE(button2_vpin) {

  relay2_state = param.asInt();

  digitalWrite(relay2_pin, relay2_state);

}

//-----

BLYNK_WRITE(button3_vpin) {

  relay3_state = param.asInt();

  digitalWrite(relay3_pin, relay3_state);

}
```



```
}

//-----

BLYNK_WRITE(button4_vpin) {

    relay4_state = param.asInt();

    digitalWrite(relay4_pin, relay4_state);

}

//-----


void setup()

{

    // Debug console

    Serial.begin(115200);

    //-----

    pinMode(button1_pin, INPUT_PULLUP);

    pinMode(button2_pin, INPUT_PULLUP);

    pinMode(button3_pin, INPUT_PULLUP);

    pinMode(button4_pin, INPUT_PULLUP);

    //-----

    pinMode(relay1_pin, OUTPUT);

    pinMode(relay2_pin, OUTPUT);

    pinMode(relay3_pin, OUTPUT);

    pinMode(relay4_pin, OUTPUT);

    //-----

    //During Starting all Relays should TURN OFF

    digitalWrite(relay1_pin, HIGH);

    digitalWrite(relay2_pin, HIGH);
```

```

digitalWrite(relay3_pin, HIGH);

digitalWrite(relay4_pin, HIGH);

//-----

Blynk.begin(auth, ssid, pass);

// You can also specify server:

//Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);

//Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);

//-----

//Blynk.virtualWrite(button1_vpin, relay1_state);

//Blynk.virtualWrite(button2_vpin, relay2_state);

//Blynk.virtualWrite(button3_vpin, relay3_state);

//Blynk.virtualWrite(button4_vpin, relay4_state);

//-----

}

void loop()

{

  Blynk.run();

  timer.run();

  // You can inject your own code or combine it with other sketches.

  // Check other examples on how to communicate with Blynk. Remember

  // to avoid delay() function!

  listen_push_buttons();

}

void listen_push_buttons(){

  //-----

```

```
if(digitalRead(button1_pin) == LOW){  
    delay(200);  
    control_relay(1);  
    Blynk.virtualWrite(button1_vpin, relay1_state); //update button state  
}  
//-----  
else if (digitalRead(button2_pin) == LOW){  
    delay(200);  
    control_relay(2);  
    Blynk.virtualWrite(button2_vpin, relay2_state); //update button state  
}  
//-----  
else if (digitalRead(button3_pin) == LOW){  
    delay(200);  
    control_relay(3);  
    Blynk.virtualWrite(button3_vpin, relay3_state); //update button state  
}  
//-----  
else if (digitalRead(button4_pin) == LOW){  
    delay(200);  
    control_relay(4);  
    Blynk.virtualWrite(button4_vpin, relay4_state); //update button state  
}  
//-----  
}  
  
void control_relay(int relay){  
    //-----
```

```
if(relay == 1){  
    relay1_state = !relay1_state;  
    digitalWrite(relay1_pin, relay1_state);  
    Serial.print("Relay1 State = ");  
    Serial.println(relay1_state);  
    delay(50);  
}  
//-----  
else if(relay == 2){  
    relay2_state = !relay2_state;  
    digitalWrite(relay2_pin, relay2_state);  
    delay(50);  
}  
//-----  
else if(relay == 3){  
    relay3_state = !relay3_state;  
    digitalWrite(relay3_pin, relay3_state);  
    delay(50);  
}  
//-----  
else if(relay == 4){  
    relay4_state = !relay4_state;  
    digitalWrite(relay4_pin, relay4_state);  
    delay(50);  
}  
}
```

Analysis

This code controls a set of four relays (electronic switches) using a Blynk connection. The Blynk app allows users to remotely control the relays through a smartphone or web interface.

Breakdown:

1. Configuration (#define statements):

- The code starts with defining various constants like Blynk template ID, device name, and authentication token (replace with your own). These are likely used for Blynk cloud registration.
- WiFi credentials (SSID and password) are also defined for the ESP32 device to connect to the internet.
- Pin assignments for four buttons and four relays are specified.
- Virtual pin numbers corresponding to the physical buttons are defined.

2. Libraries:

- The code includes libraries for WiFi communication (WiFi.h, WiFiClient.h) and the Blynk connection (BlynkSimpleEsp32.h).

3. Blynk Functions:

- BLYNK_CONNECTED(): This function runs when the ESP32 connects to Blynk. It synchronizes the virtual pin states between the device and the Blynk app.
- BLYNK_WRITE(Vpin): These are individual functions for each virtual pin (V1 to V4). They are triggered whenever a button state changes in the Blynk app. The function reads the new state (param.asInt()) and updates the corresponding relay state (relay1_state to relay4_state). It then turns the relay on or off based on the received state using digitalWrite.

4. Setup Function:

- Initializes serial communication for debugging.
- Sets all button pins as inputs with pull-up resistors to handle button presses.
- Sets all relay pins as outputs.
- Initializes the Blynk connection with the defined credentials.
- Optionally sends the initial relay states to Blynk (commented out).

5. Loop Function:

- Calls Blynk.run() to maintain the Blynk connection.
- Calls a timer function (not shown in the provided code).
- Calls the listen_push_buttons() function to monitor button presses.

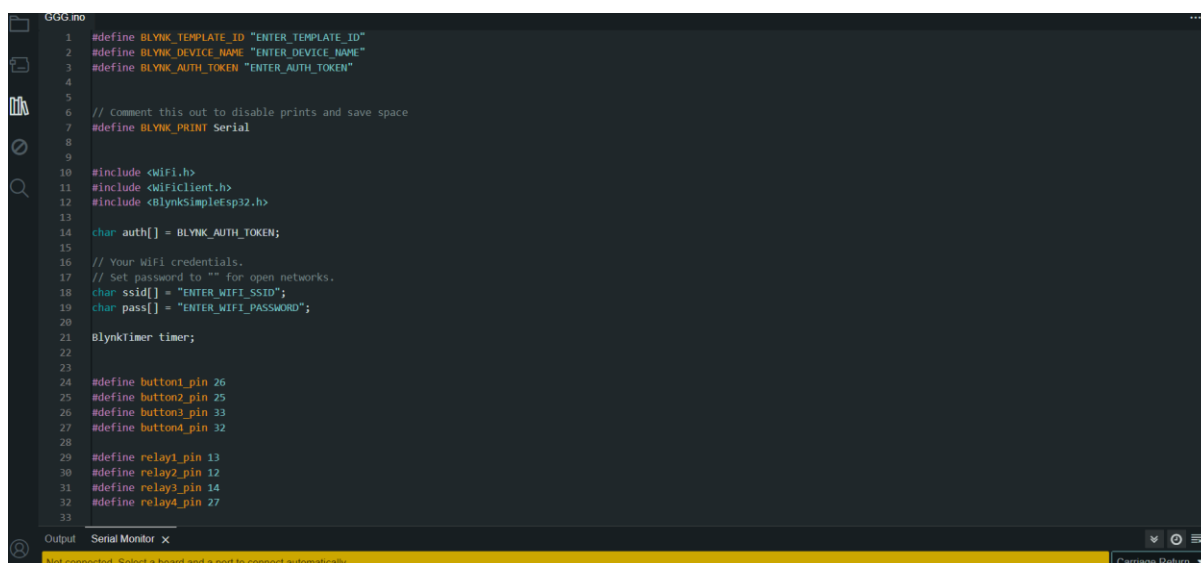
6. listen_push_buttons():

- This function continuously checks the state of each button using digitalRead().
- If a button is pressed (LOW state), it introduces a debounce delay (200ms with delay()) to avoid multiple triggers on a single press.
- It then calls control_relay() with the corresponding relay number (1 to 4).
- Finally, it updates the virtual pin state in the Blynk app using Blynk.virtualWrite().

7. control_relay(int relay):

- This function takes a relay number (1 to 4) as input.
- It toggles the state of the corresponding relay variable (relay1_state to relay4_state).
- It then writes the new state to the actual relay pin using digitalWrite.

3.2 SNAPSHOTS OF CODING AND RESULTS



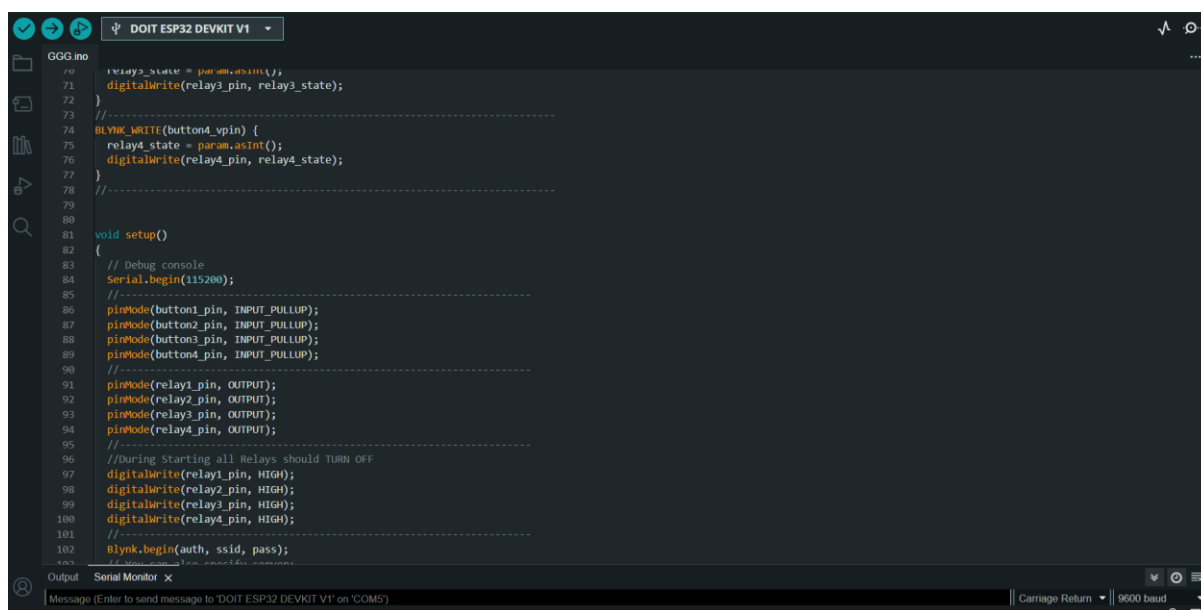
```

1  #define BLYNK_TEMPLATE_ID "ENTER_TEMPLATE_ID"
2  #define BLYNK_DEVICE_NAME "ENTER_DEVICE_NAME"
3  #define BLYNK_AUTH_TOKEN "ENTER_AUTH_TOKEN"
4
5
6  // Comment this out to disable prints and save space
7  #define BLYNK_PRINT Serial
8
9
10 #include <WiFi.h>
11 #include <WiFiClient.h>
12 #include <BlynkSimpleEsp32.h>
13
14 char auth[] = BLYNK_AUTH_TOKEN;
15
16 // Your WiFi credentials.
17 // Set password to "" for open networks.
18 char ssid[] = "ENTER_WIFI_SSID";
19 char pass[] = "ENTER_WIFI_PASSWORD";
20
21 BlynkTimer timer;
22
23
24 #define button1_pin 26
25 #define button2_pin 25
26 #define button3_pin 33
27 #define button4_pin 32
28
29 #define relay1_pin 13
30 #define relay2_pin 12
31 #define relay3_pin 14
32 #define relay4_pin 27
33

```

Output Serial Monitor x

Not connected. Select a board and a port to connect automatically.



```

70 relay3_state = param.asInt();
71 digitalWrite(relay3_pin, relay3_state);
72 }
73 //-----
74 BLYNK_WRITE(button4_vpin) {
75   relay4_state = param.asInt();
76   digitalWrite(relay4_pin, relay4_state);
77 }
78 //-----
79
80
81 void setup()
82 {
83   // Debug console
84   Serial.begin(115200);
85   //-----
86   pinMode(button1_pin, INPUT_PULLUP);
87   pinMode(button2_pin, INPUT_PULLUP);
88   pinMode(button3_pin, INPUT_PULLUP);
89   pinMode(button4_pin, INPUT_PULLUP);
90   //-----
91   pinMode(relay1_pin, OUTPUT);
92   pinMode(relay2_pin, OUTPUT);
93   pinMode(relay3_pin, OUTPUT);
94   pinMode(relay4_pin, OUTPUT);
95   //-----
96   //During Starting all Relays should TURN OFF
97   digitalWrite(relay1_pin, HIGH);
98   digitalWrite(relay2_pin, HIGH);
99   digitalWrite(relay3_pin, HIGH);
100  digitalWrite(relay4_pin, HIGH);
101  //-----
102  Blynk.begin(auth, ssid, pass);
103  //-----

```

Output Serial Monitor x

Message (Enter to send message to 'DOIT ESP32 DEVKIT V1' on 'COM5')

Carriage Return 9600 baud

4 CONCLUSIONS

The project effectively demonstrates a remote control system for four relays using the Blynk IoT platform. The Blynk connection allows users to control the relays via a smartphone app or web interface. The code incorporates essential functionalities like debouncing for button presses, efficient communication with Blynk, and clear relay state management.

4.1 INFERENCE

This system can be further enhanced by implementing features like:

- **Two-way communication:** The code can be extended to receive sensor data from the ESP32 device and display it on the Blynk app, enabling real-time monitoring.
- **Security measures:** Consider incorporating user authentication or encryption mechanisms to protect against unauthorized access to the relays.
- **Scheduled operations:** Allow users to define schedules or timers for automatic relay control within the Blynk app.

By incorporating these improvements, the system can become more versatile and secure for various remote-control applications.

4.2 FUTURE WORK

Improved User Experience:

- **Mobile App Development:** Design a custom mobile application for Blynk that provides a more intuitive and user-friendly interface for controlling the relays and viewing sensor data (if implemented).
- **Web Interface Enhancements:** Develop a dedicated web interface for the system, allowing users to control relays and view data from a web browser on any device.
- **Notifications and Alerts:** Implement a notification system within the Blynk app to alert users about specific events, such as sensor readings exceeding thresholds or relay malfunctions.

Security and Reliability:

- **User Authentication:** Integrate user login functionality to restrict unauthorized access to the relay control system.
- **Data Encryption:** Implement encryption for communication between the ESP32 device and the Blynk cloud to protect sensitive data from interception.
- **Error Handling and Recovery:** Develop error handling routines to gracefully handle unexpected situations like network outages or sensor malfunctions. Consider incorporating fail-safe mechanisms for the relays in case of errors.

Hardware Integration:

- **Additional Relays:** Expand the system to control more relays by adding extra relay modules and modifying the code accordingly.
- **Sensor Integration:** Integrate various sensors (temperature, humidity, light, etc.) to gather environmental data and use it for control logic or data visualization.
- **Display Integration:** Connect a display (LCD or OLED) to the ESP32 to provide local feedback on relay states, sensor readings, or other relevant information.

Enhanced Functionality:

- ☐ **Scheduling and Timers:** Integrate scheduling features within the Blynk app. This allows users to define automatic on/off cycles for the relays based on specific times or sensor readings.
- ☐ **Advanced Control Logic:** Develop more complex control logic within the code. For example, relays could be programmed to turn on/off based on combinations of button presses, sensor readings, or time-based triggers.

REFERENCES (IN IEEE FORMAT)

LIST OF PUBLICATIONS

INTERNATIONAL JOURNALS

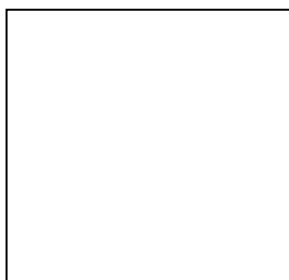
- R. Niranjana, A. S, V. M and V. S, "Effectual Home Automation using ESP32 Node MCU," 2022 International Conference on Automation, Computing and Renewable Systems (ICACRS), Pudukkottai, India, 2022, pp. 1-5, Doi: 10.1109/ICACRS55517.2022.10028992.
- Pujari, Uma and Patil, Prasenjeet and Bahadure, Nilesh and Asnodkar, Manvita, Internet of Things based Integrated Smart Home Automation System (May 1, 2020). 2nd International Conference on Communication & Information Processing (ICCIP) 2020, Available at SSRN: <https://ssrn.com/abstract=3645458> or <http://dx.doi.org/10.2139/ssrn.3645458>

BIODATA

Name : Abhisha Chakrabarti
Mobile Number : 9163752279
E-mail : Abhisha.chakrabarti2022@vitstudent.ac.in
Permanent Address : Regent Enclave, VIP road, Kaikhali, Kolkata-



Name : Ralf Victor
Mobile Number : 8754895610
E-mail : ralfpaul.victor2022@vitstudent.ac.in
Permanent Address : Goodwill Apartment, Nerul, Navi Mumbai - 400706



Name :
Mobile Number :
E-mail :
Permanent Address: